COSC 3319-01 Lab 3 Report

Name: Darren William Lehmann

Course & Section: COSC 3319-01

Meeting Days: MWF 8:00-8:50am

Submission Date: November 30, 2025

Grading Option Completed: B

Contents

1 Option C — Relative File

1.1 Implementation Overview

1.2 Hash Functions

1.2.1 MyHash (Bad Hash)

The MyHash function provided is intentionally poor. It only uses a few characters from the key, which means that keys that differ outside the positions checked by the hash could produce the same hash, creating a collision. The float conversion, only to convert back to int at the end, seems unnecessary, and the constant divisors 517, 217, and 256 do not help spread the values out; they just scale the sum.

```
// === MY HASH (given; intentionally poor) ===
func MyHash(key string, mod int) int {
    a := asciiCat(key, 1, 1) + asciiCat(key, 5, 5)
    b := asciiCat(key, 3, 4)
    c := asciiCat(key, 5, 6)
    h := float64(a)/517.0 + float64(b)/217.0 + float64(c)/256.0
    if h < 0 {
        h = -h
    }
    return int(h) % mod
}
```

Listing 1: MyHash implementation

1.3 Insertion and Collision Handling

The insertion logic starts at the hashed index. If the slot is free, it inserts the key into that slot. If the slot is full, it uses the prober to find the next available slot. In linear probing, it will check the start slot plus one. This process repeats until an empty slot is found, and the probe counter is incremented to show how many attempts it took.

```
func InsertKeyC(tableFile *os.File, key []byte, start int, prober
   Prober) (int, int, error) {
    prober.Init(start, int(TableSize))
    idx := start
    probeCount := 0

    for probeCount < TableSize {
        var slot Slot
        if err := ReadSlot(tableFile, idx, &slot); err != nil {
            return -1, probeCount, err
        }

        if slot.Filled == 0 {
            copy(slot.Key[:], key)
            slot.Orig = int32(start)
            slot.Final = int32(idx)
            slot.Probes = int32(probeCount + 1)
            slot.Filled = 1
            if err := WriteSlot(tableFile, idx, &slot); err != nil {
                return -1, probeCount, err
            }
```

```
            return idx, probeCount + 1, nil
        }

        idx = prober.Next()
        probeCount++
    }

    return -1, probeCount, fmt.Errorf("table full")
}
```

Listing 2: Linear probing insertion for 75 keys

1.4 Query Testing and Probe Statistics

To evaluate the hash function I gathered the min,max, and average of first 25 keys and the last 25 keys. I also compared the average of all 75 keys to the theoretical average.

Table 1: Probe Statistics for Linear Probing

Hash	Probe	F25 (min/max/avg)	L25 (min/max/avg)	All75 (avg)	Theory (avg)
MyHash	Linear	1/22/5.65	1/27/3.08	3.92	2.50

1.5 Theoretical vs. Measured Analysis

Given:

$$n = 75, \quad m = 100, \quad \alpha = \frac{n}{m} = 0.75$$

Successful Search (Linear Probing):

$$E_s = \frac{1}{2}\left(1 + \frac{1}{1-\alpha}\right)$$

Evaluation with $\alpha = 0.75$:

$$E_s = \frac{1}{2}\left(1 + \frac{1}{1-0.75}\right) = \frac{1}{2}\left(1 + \frac{1}{0.25}\right) = \frac{1}{2}(1+4) = \frac{5}{2} = 2.5$$

Comparison with Theory:

Table 2: Measured vs. Theoretical Probes (Linear Probing, Successful Search)

Search Set	Measured Avg Probes	Theoretical Avg Probes
First 25 keys	5.65	2.5
Last 25 keys	3.08	2.5
All 75 keys	3.92	2.5

1.5.1 Explanation

The measured probes were higher than the theoretical average. This shows that the hash function used did a poor job at spreading the 75 keys over 100 slots. The first 25 took almost double the amount of probes on average than the theoretical average, while the last 25 were closer to the theoretical average.

1.6 Table Dump Output

1.6.1 Complete Table Display (Slots 0–99)

```
=== Table Dump ===
Hash address    Contents at address  Original hash   Final hash  Probes
-------------------------------------------------------------------------

0               G4iCoa3vqcMNXypt     92              0               9
1               P6z6lJg68Qi4Qh5a     99              1               3
2               w6taYjy6apLID6VM     2               2               1
3               XcrcG2aBIxVvowUx     93              3               11
4               VIs5gBW2Du5KKCB3     94              4               11
5               GnNfs99CyXtt8ltD     5               5               1
6               03oYAv9FHj3ntES3     6               6               1
7               rw25zuE6gQTcR9MQ     0               7               8
8               Rss0bMjTMdBCJRDi     92              8               17
9               URAq0d4iOtiCKJxf     88              9               22
10              STx0Ae5OEGmfPiB1     10              10              1
11              M6rkIdzXL1v5wSXU     11              11              1
12              6PhrLOLEoSFUw12o     10              12              3
13              771YcuUELnP1gS1X     10              13              4
14              (empty)
15              (empty)
16              (empty)
17              (empty)
18              zTBmPxZPqwcj4bYJ     18              18              1
19              (empty)
20              (empty)
21              (empty)
22              (empty)
23              NodxCgQ10vtqkufT     23              23              1
24              MOGNdko4Wh8YNxFa     24              24              1
25              gGrsLvp1p9HefRNM     23              25              3
26              71ipeOg2kJPifs3R     24              26              3
27              (empty)
28              (empty)
29              (empty)
30              (empty)
31              (empty)
32              vIThpRpZt6T2WFDk     32              32              1
33              (empty)
34              (empty)
35              (empty)
36              (empty)
37              Tmt2bzgUTrYh7L2R     37              37              1
38              v8MzHs6i7quhlbxf     37              38              2
39              SPlrhCZhHiUw3eIy     39              39              1
40              (empty)
41              (empty)
42              RNczIjLhcT9nVZFb     42              42              1
43              rL0w62UqvoaJPCKP     43              43              1
44              AhZjGGhfRGWWeatr     43              44              2
45              uKqy42RdNMA5YnOm     42              45              4
46              (empty)
47              (empty)
48              (empty)
49              (empty)
```

```
50                (empty)
51                B52h35qGLCwyUUFy     51              51              1
52                9o2d33dfNhkYEGd8     51              52              2
53                FLHP30tA4E8ZKqxd     53              53              1
54                RlPXjwkO2vZpNLLU     52              54              3
55                X34HTtKZWpig4xO9     53              55              3
56                5PMDQuWDBaXXMDs5     52              56              5
57                GZEicTpCdI5xndZJ     57              57              1
58                JaS41YZliYCfDAss     58              58              1
59                aoH1A69ayQtXkeks     59              59              1
60                6Mcx4fNznFLRNxVo     60              60              1
61                c1SC9LFErx0Ikdyh     61              61              1
62                Od9bTmGRYUBxZvWV     55              62              8
63                WT7SYUmqgyEV2JbU     61              63              3
64                PcHaDSFBYQA9eDwa     60              64              5
65                0V2DjJdgRcnW1xge     65              65              1
66                t8tpKalT0yXZ6BtP     65              66              2
67                6FEWS3y9nd25BZg7     65              67              3
68                pf3tLUoKaJhoHkSE     65              68              4
69                p2M6T9vzfJuy5wP3     69              69              1
70                Ly8uLp270ZWQ3Slt     56              70              15
71                SjSVMCNUKJ9HaSeK     69              71              3
72                lzO1LDvBMtP3RV7n     66              72              7
73                Pl3tCYqjSgW2gwNs     62              73              12
74                gJKD06MXSRW7STBJ     54              74              21
75                QMqSlhMrQOfxTJ8M     75              75              1
76                oFupWNQhKXcMVZYV     74              76              3
77                7eEJiQNYk42EyCYP     73              77              5
78                f7NRjXKkkD7LqocM     78              78              1
79                pkocGBDOMUOelJMx     79              79              1
80                IWXDcG7iwFaO6Kki     80              80              1
81                X0ZGVmeodkS31VKb     78              81              4
82                K6N1rdL158Avqc81     82              82              1
83                TW4jpwQlbQWx6K0A     78              83              6
84                OssRNRh0eMJVA2Jr     84              84              1
85                r58pD6asUZfqXMiZ     85              85              1
86                oX95VTkjrPk6lnXs     60              86              27
87                (empty)
88                FqcKjZ1Fmd0A7dbP     88              88              1
89                QRQ0Zsr7qv4mZuPw     89              89              1
90                1mwhjQD4093NIWs2     90              90              1
91                zER1ZfRtxewEgarL     90              91              2
92                BEcfZIGK0MGfyCav     92              92              1
93                S4iEWrIbYU00XPEC     89              93              5
94                Pkc3Yo6dGHvsT6Tc     94              94              1
95                zVlsLvy4RCCTayyM     95              95              1
96                1Uv5Wp2c1eKyXZxz     95              96              2
97                xC7gnZz00wKqn3bI     97              97              1
98                sscXzYpal3LweUyP     94              98              5
99                JBxbnCbzRgHxtvaL     99              99              1
```

Listing 3: Full table dump for Option C

2 Option B — Memory Table

2.1 Implementation Overview

2.2 Array-Based Data Structure Design

2.2.1 Memory Layout and Slot Organization

2.2.2 Constraints (No Maps on Hashing Path)

2.3 Hash Functions

2.3.1 YourHash (Redesigned Hash)

```go
func YourHash(key string, mod int) int {
    h := fnv.New32a()
    h.Write([]byte(key))
    return int(h.Sum32() % uint32(mod))
}
```

Listing 4: YourHash implementation in memory

The YourHash That is provided improved upon MyHash by using FNV-1a over 16 byte keys. YourHash creates a sum with all 16 bytes that can then spread the 75 keys uniformly over the 100 slots with less collisions. With fewer collisions, the insertion algorithm performs fewer probe loops, which makes the overall insertion more efficient.

2.4 Insertion and Collision Handling

```go
func InsertKeyB(memTable []MemSlot, key []byte, start int, prober
   Prober) (int, int, error) {
    prober.Init(start, M)
    idx := start
    probeCount := 0

    for probeCount < M {
        slot := &memTable[idx] // pointer to memTable

        if !slot.Filled {
            copy(slot.Key[:], key)
            slot.Orig = start
            slot.Final = idx
            slot.Probes = probeCount + 1
            slot.Filled = true
            return idx, probeCount + 1, nil
        }

        idx = prober.Next()
        probeCount++
    }
    return -1, probeCount, fmt.Errorf("table full")
}
```

Listing 5: Linear probing insertion for 75 keys in memory

2.5 Query Testing and Probe Statistics

Hash	Probe	F25 (min/max/avg)	L25 (min/max/avg)	All75 (avg)	Theory (avg)
YourHash	Linear	1/5/1.61	1/4/1.57	1.88	2.50

2.6 Table Dump Output

2.6.1 Complete Array Display (Indices 0–99)

```
=== Table Dump ===
Hash address    Contents at address  Original hash   Final hash  Probes
-------------------------------------------------------------------------

0               GZEicTpCdI5xndZJ     0               0               1
1               JBxbnCbzRgHxtvaL     0               1               2
2               pf3tLUoKaJhoHkSE     2               2               1
3               w6taYjy6apLID6VM     2               3               2
4               (empty)
5               (empty)
6               (empty)
7               G4iCoa3vqcMNXypt     7               7               1
8               (empty)
9               (empty)
10              6Mcx4fNznFLRNxVo     10              10              1
11              Rss0bMjTMdBCJRDi     11              11              1
12              (empty)
13              (empty)
14              VIs5gBW2Du5KKCB3     14              14              1
15              Pkc3Yo6dGHvsT6Tc     15              15              1
16              B52h35qGLCwyUUFy     16              16              1
17              1Uv5Wp2c1eKyXZxz     16              17              2
18              03oYAv9FHj3ntES3     18              18              1
19              771YcuUELnP1gS1X     17              19              3
20              t8tpKalT0yXZ6BtP     20              20              1
21              NodxCgQ10vtqkufT     21              21              1
22              zTBmPxZPqwcj4bYJ     20              22              3
23              pkocGBDOMUOelJMx     23              23              1
24              SjSVMCNUKJ9HaSeK     20              24              5
25              K6N1rdL158Avqc81     25              25              1
26              Od9bTmGRYUBxZvWV     26              26              1
27              sscXzYpal3LweUyP     25              27              3
28              r58pD6asUZfqXMiZ     28              28              1
29              RNczIjLhcT9nVZFb     27              29              3
30              X34HTtKZWpig4xO9     30              30              1
31              X0ZGVmeodkS31VKb     30              31              2
32              WT7SYUmqgyEV2JbU     30              32              3
33              IWXDcG7iwFaO6Kki     33              33              1
34              QRQ0Zsr7qv4mZuPw     27              34              8
35              lzO1LDvBMtP3RV7n     34              35              2
36              5PMDQuWDBaXXMDs5     36              36              1
37              MOGNdko4Wh8YNxFa     37              37              1
38              AhZjGGhfRGWWeatr     37              38              2
39              (empty)
40              Tmt2bzgUTrYh7L2R     40              40              1
41              TW4jpwQlbQWx6K0A     40              41              2
42              f7NRjXKkkD7LqocM     42              42              1
43              OssRNRh0eMJVA2Jr     43              43              1
```

```
44                  c1SC9LFErx0Ikdyh        44                  44                  1
45                  (empty)
46                  (empty)
47                  BEcfZIGK0MGfyCav        47                  47                  1
48                  zVlsLvy4RCCTayyM        47                  48                  2
49                  p2M6T9vzfJuy5wP3        48                  49                  2
50                  rw25zuE6gQTcR9MQ        47                  50                  4
51                  URAq0d4iOtiCKJxf        51                  51                  1
52                  oX95VTkjrPk6lnXs        47                  52                  6
53                  JaS41YZliYCfDAss        53                  53                  1
54                  71ipeOg2kJPifs3R        53                  54                  2
55                  QMqSlhMrQOfxTJ8M        55                  55                  1
56                  6PhrLOLEoSFUw12o        55                  56                  2
57                  uKqy42RdNMA5YnOm        55                  57                  3
58                  gJKD06MXSRW7STBJ        56                  58                  3
59                  xC7gnZz00wKqn3bI        59                  59                  1
60                  PcHaDSFBYQA9eDwa        59                  60                  2
61                  (empty)
62                  1mwhjQD4093NIWs2        62                  62                  1
63                  zER1ZfRtxewEgarL        62                  63                  2
64                  Ly8uLp270ZWQ3Slt        62                  64                  3
65                  FqcKjZ1Fmd0A7dbP        62                  65                  4
66                  v8MzHs6i7quhlbxf        65                  66                  2
67                  7eEJiQNYk42EyCYP        62                  67                  6
68                  (empty)
69                  (empty)
70                  (empty)
71                  aoH1A69ayQtXkeks        71                  71                  1
72                  vIThpRpZt6T2WFDk        72                  72                  1
73                  M6rkIdzXL1v5wSXU        73                  73                  1
74                  9o2d33dfNhkYEGd8        73                  74                  2
75                  gGrsLvp1p9HefRNM        72                  75                  4
76                  oFupWNQhKXcMVZYV        73                  76                  4
77                  STx0Ae5OEGmfPiB1        77                  77                  1
78                  (empty)
79                  (empty)
80                  (empty)
81                  RlPXjwkO2vZpNLLU        81                  81                  1
82                  SPlrhCZhHiUw3eIy        82                  82                  1
83                  (empty)
84                  6FEWS3y9nd25BZg7        84                  84                  1
85                  (empty)
86                  (empty)
87                  (empty)
88                  (empty)
89                  P6z6lJg68Qi4Qh5a        89                  89                  1
90                  FLHP30tA4E8ZKqxd        90                  90                  1
91                  GnNfs99CyXtt8ltD        89                  91                  3
92                  (empty)
93                  (empty)
94                  rL0w62UqvoaJPCKP        94                  94                  1
95                  Pl3tCYqjSgW2gwNs        95                  95                  1
96                  S4iEWrIbYU00XPEC        96                  96                  1
97                  0V2DjJdgRcnW1xge        97                  97                  1
98                  (empty)
99                  Xcrcg2aBIxVvowUx        99                  99                  1
```

Listing 6: Full memory table dump for Option B

2.7 Theoretical vs. Measured Comparison

Search Set	Measured Avg Probes C	Measured Avg Probes B	Theoretical
First 25 keys	5.65	1.61	2.5
Last 25 keys	3.08	1.57	2.5
All 75 keys	3.92	1.88	2.5

2.8 Conclusion

Together, Option B and Option C show the clear difference between the two hash functions. MyHash was intentionally poor, while YourHash was a better implementation. MyHash produced higher average probes compared to the theoretical average, while YourHash stayed below the theoretical average. Both options used the same table size, the same keys, and the same linear probing logic, but Option C's hash required roughly twice as many probes on average as the hash used in Option B.

A well-designed hash keeps probe lengths short and makes access patterns more efficient, while a poorly designed hash leads to clustering and an inefficient number of probes.